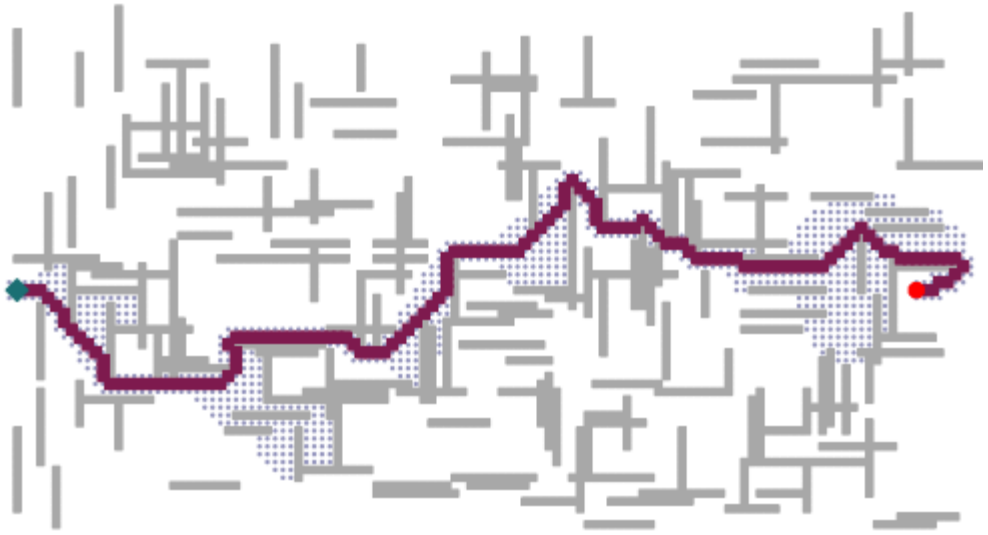
Local Search

ITI0210, lecture 4 (2021)

Review

Heuristic search: we either

- Know the exact goal state
- Recognize the goal once we see it



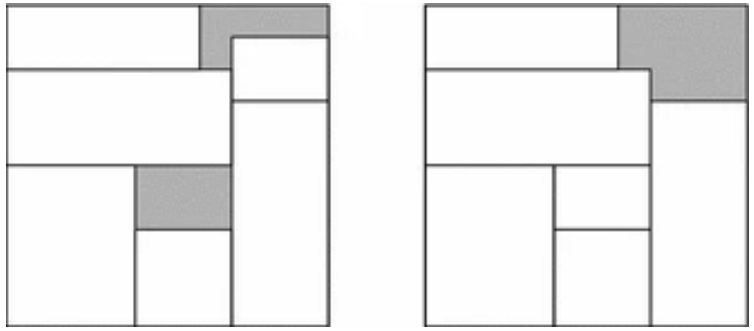
Solution is a sequence of actions

Optimization Problems

What if we **don't know** what the goal state is?

Examples:

- Logistics: optimal delivery van assignment
- Industry: cut parts from material with least waste
- Machine learning: optimal weights for neural network
- Planning: timetable for lectures



Kierkosz, I., Luczak, M. A hybrid evolutionary algorithm for the two-dimensional packing problem. *Cent Eur J Oper Res* **22,** 729–753 (2014)

Optimization Problems

Many problems where we need the optimal state

the steps to it are usually unimportant

Path planning vs optimization

	Rubik's Cube	Lecture timetable
Final state	Known 	Timetable with least "holes", very long days, etc
How to get there	The whole point of the puzzle	Don't care

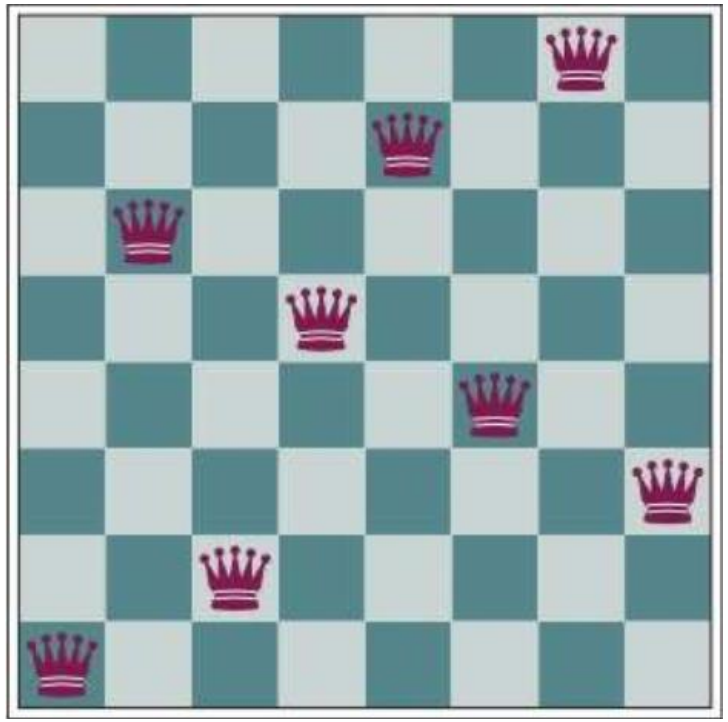
The N-Queens Problem

Toy Problem to study optimization (and for your homework)

N-Queens

NxN board

place N chess queens so that none can “capture” others



N-Queens

Solution idea:

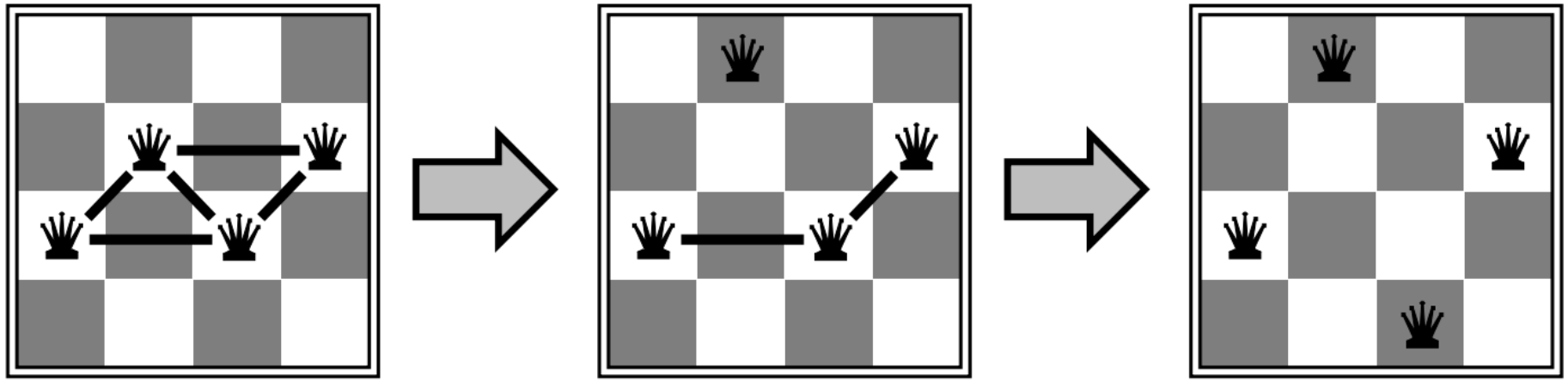
1. Put all the queens on the board (initial solution, not correct)
2. Start moving them around, trying to improve

Question: how does the computer know the solution “improved”?

N-Queens

Value function:

How good is the current solution



h - number of “conflicting” pairs. We want $h = 0$

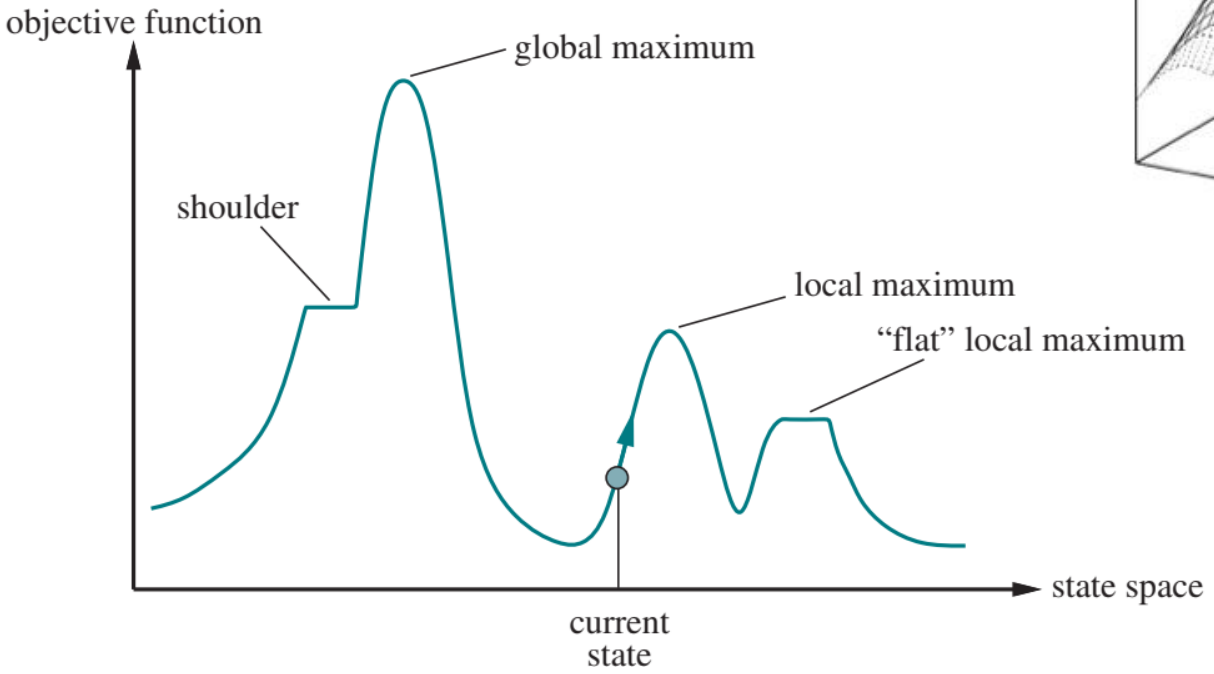
The (minimizing) hill-climbing algorithm

```
def hill_climbing(pos):
    curr_value = pos.value()
    while True:
        move, new_value = pos.best_move()
        if new_value >= curr_value:
            # no improvement, give up
            return pos, curr_value
        else:
            # position improves
            curr_value = new_value
            pos.make_move(move)
```

best_move() –
returns the move leading to lowest value neighbor state

Some search landscapes

Maximization (opposite problem)



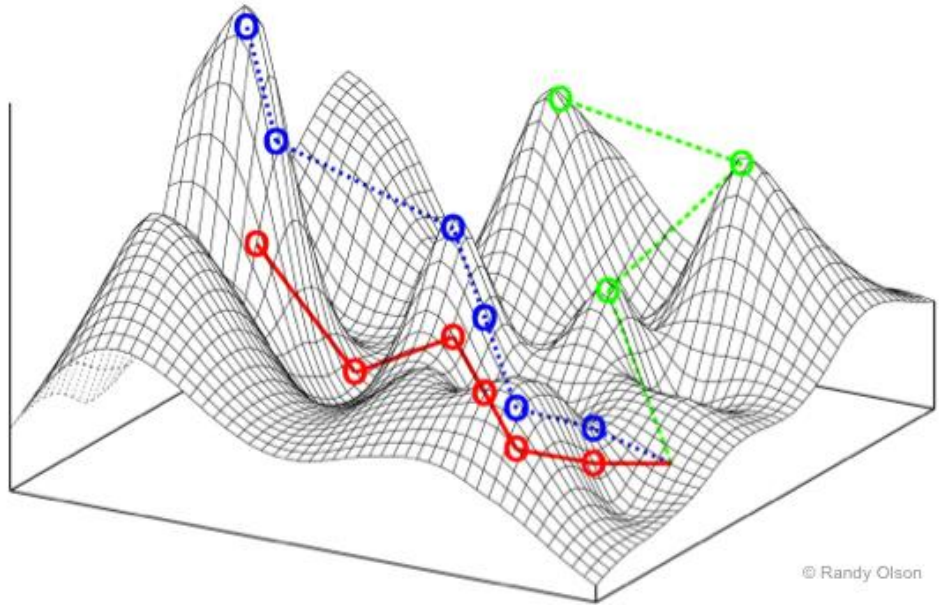
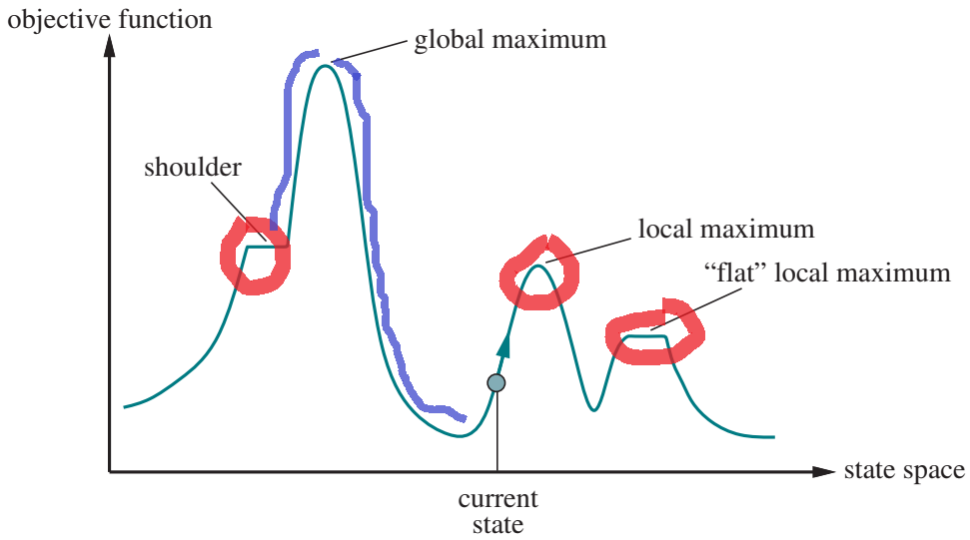


Image: Randy Olson, Wikimedia Commons

Local maxima/minima

Hill climbing finds local maxima/minima
and gets “stuck” there

Main idea of improved algorithms:
escaping local optimum points



(Maximizing) HC gets stuck in the marked points

Blue line shows starting locations where global max is findable

The Travelling Salesperson Problem

A Classic

TSP

Find a **shortest** tour passing **each** town **once**.

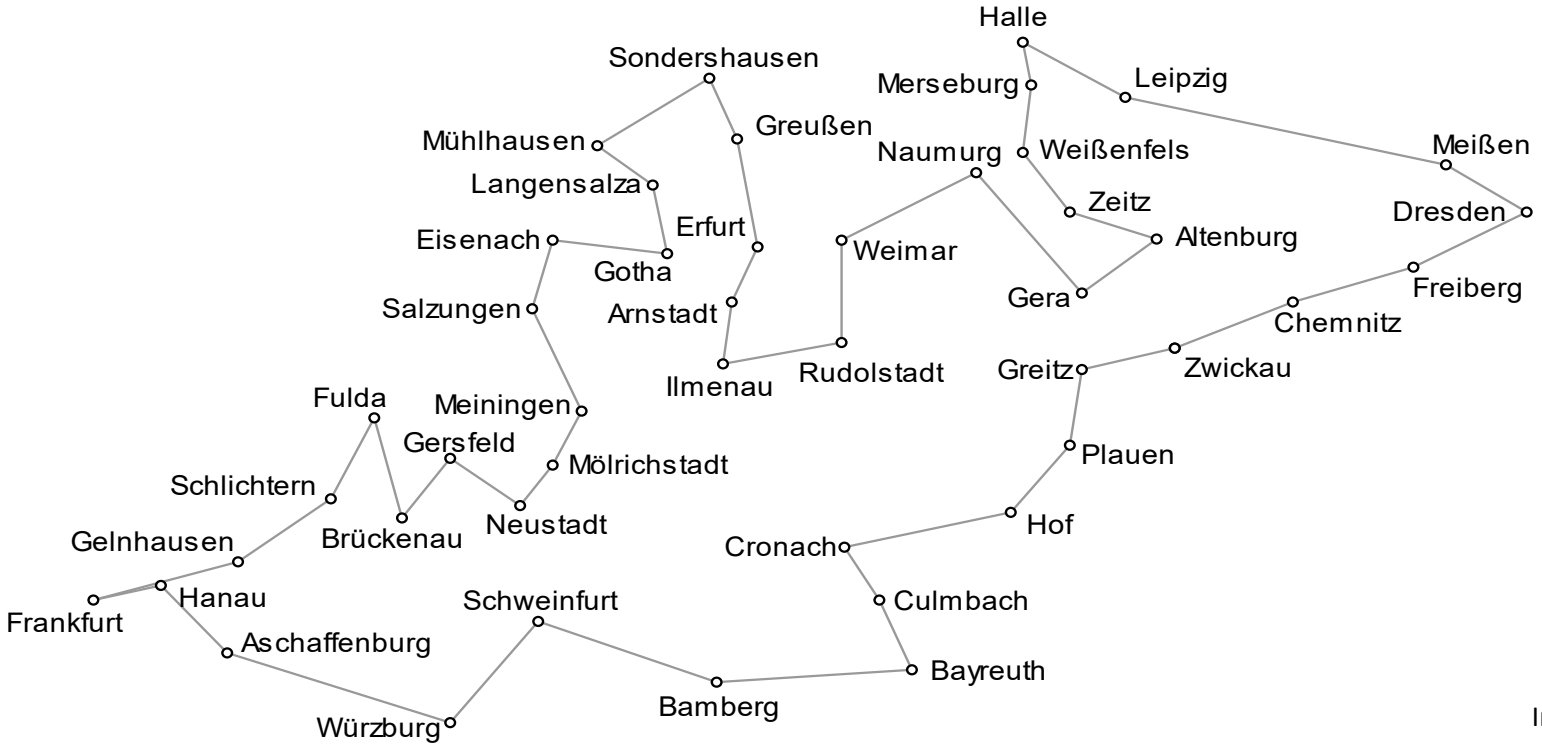


Image: Wikimedia Commons

TSP

Constructing the initial solution:

- Make random connections
- Greedy construction

  (pick a town. Join it with closest unconnected neighbor. Repeat)

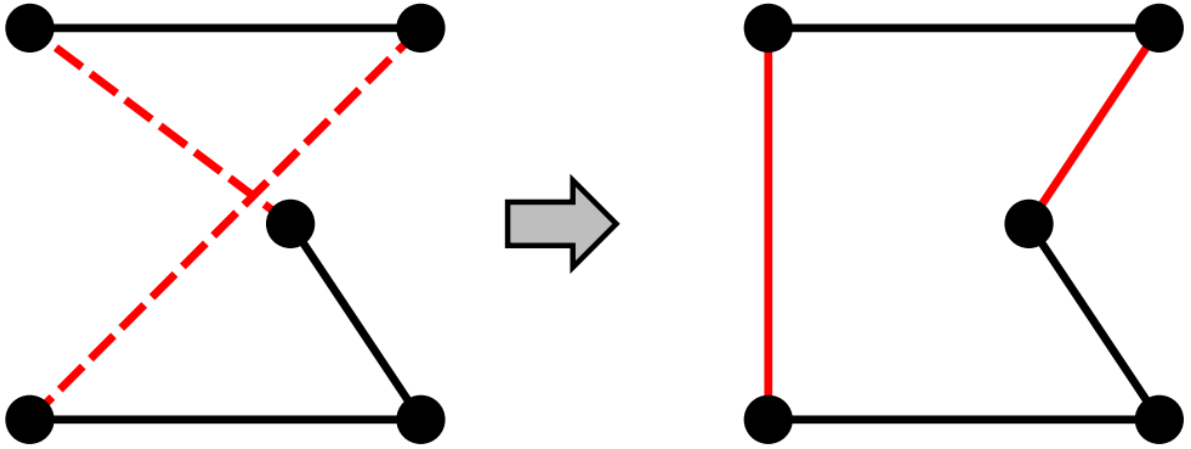
Greedy construction animated:

https://en.wikipedia.org/wiki/Travelling_salesman_problem#Heuristic_and_approximation_algorithms

TSP

Improving the solution: the 2-opt move

disconnect 2 edges, rejoin the other way



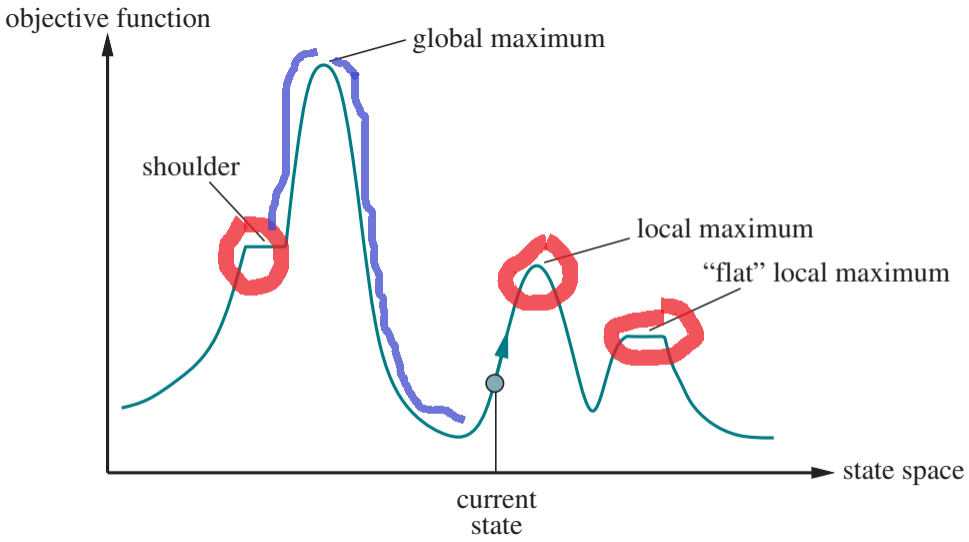
Others are 3-opt (generally λ-opt); Lin-Kernighan etc

(can read yourself, not required in course)

Improved Algorithms

Apply to N-Queens, TSP and many more

Random-restart hill climbing



Restart hill climbing multiple times from random state

Idea: may end up in blue area, global optimum neighborhood

Simulated Annealing (minimizing version)

```
def simulated_annealing(pos, schedule):
    curr_value = pos.value()
    time = 0
    while True:
        T = schedule[time]  # temperature
        if T == 0:
            return pos, curr_value
        move, new_value = pos.random_move()
        delta = new_value - curr_value
        if delta < 0:  # improvement, always OK
            curr_value = new_value
            pos.make_move(move)
        elif random.random() < math.exp(-delta/T):
            # allow bad moves, decreasing probability
            curr_value = new_value
            pos.make_move(move)
        time += 1
```

One random neighbor per iteration

"Bad"moves allowed
probability decreases with time

$$p(x) = e^{\frac{-\Delta}{T}}$$

x - state

SA Cooling Schedule

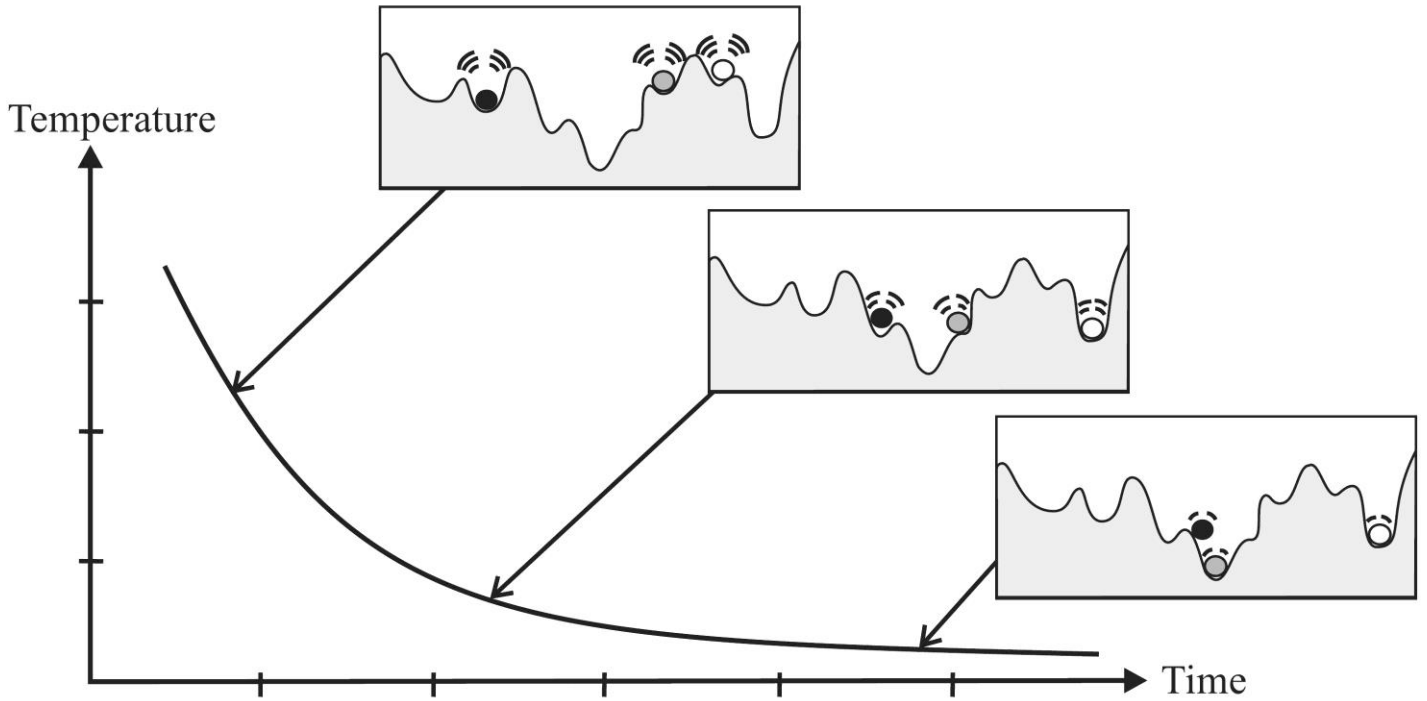


Image: Sergio Ledesma, Jose Ruiz and Guadalupe Garcia (August 29th 2012). Simulated Annealing Evolution, Simulated Annealing - Advances, Applications and Hybridizations, Marcos de Sales Guerra Tsuzuki, IntechOpen

SA Theoretical Property

Probability of being in state x ($v(x)$ – value of state; fixed T):

$p(x) = \alpha e^{-\frac{v(x)}{kT}}$ (Boltzmann distribution)

Relative probabilities of best state vs any other state

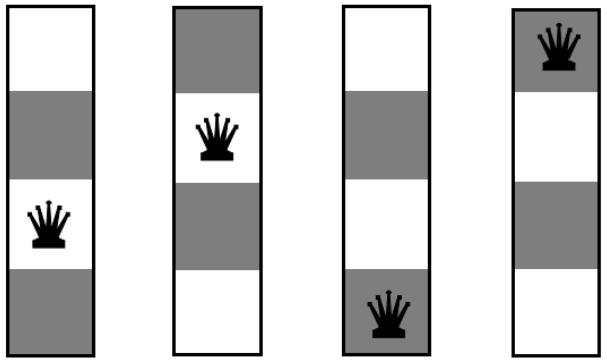
$$\frac{p(x_{best})}{p(x)} = e^{\frac{v(x)-v(x_{best})}{kT}} > 1$$

Small T and big difference to $v(x)$ ➔ profit

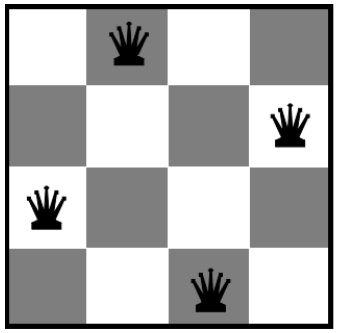
What about random-restart HC?

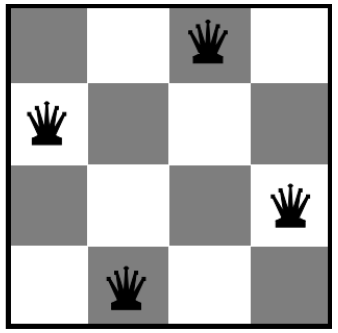
Need to arrange these pieces:

$4! = 24$ permutations



Legal solutions (2):



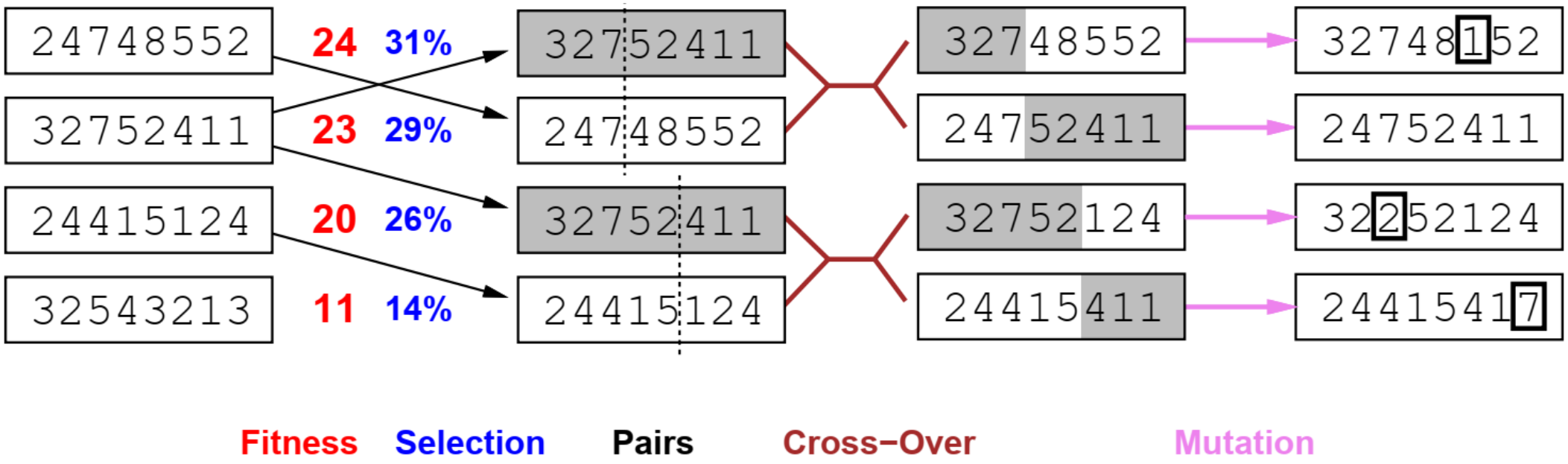


Probability that 100 random restarts produce a solution immediately:

$$1 - \left(1 - \frac{2}{24}\right)^{100} = 0.999$$

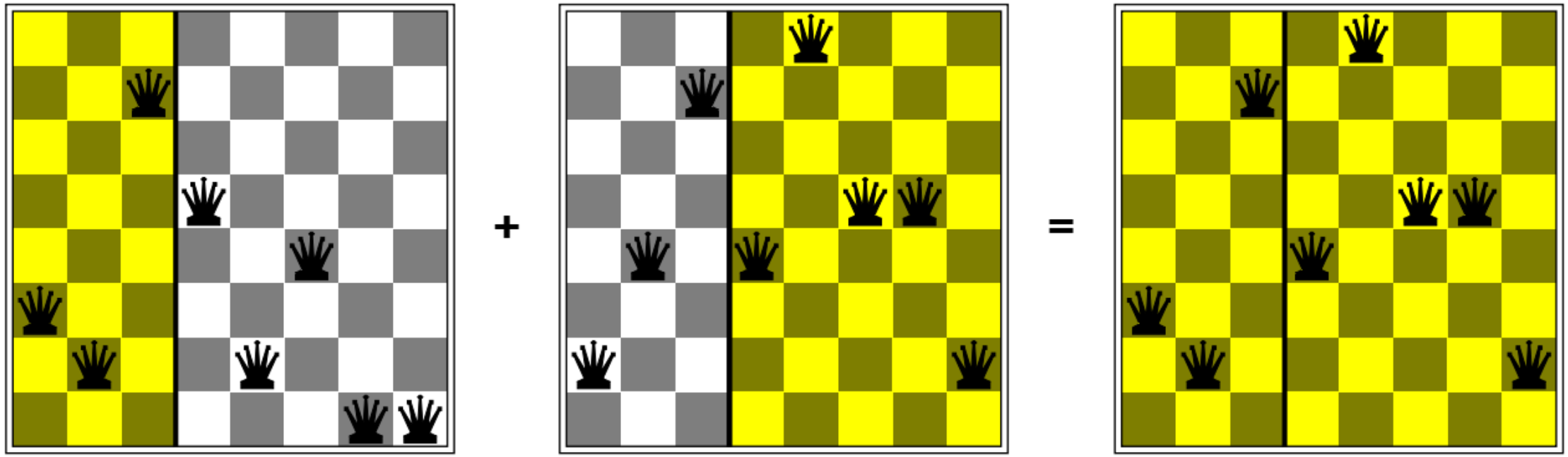
Genetic Algorithm

Population of solutions; combine them and mutate. Repeat.

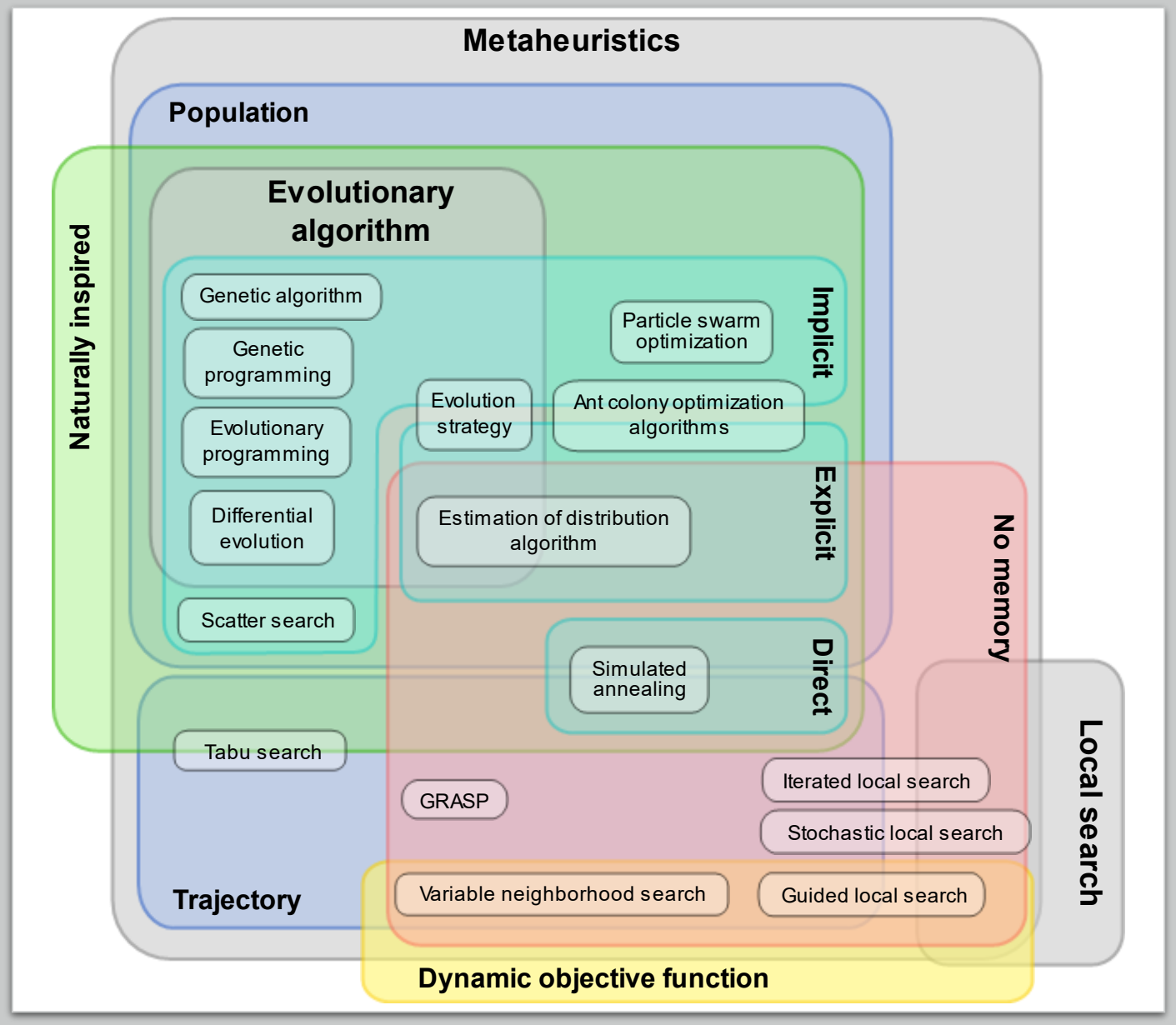


Genetic Algorithm

Crossover between two states:



Zoo of Metaheuristics



Metaheuristics

Read this paper if you:

- Care about optimization algorithms
- Are going to study for a PhD

Sörensen, Kenneth. "Metaheuristics—the metaphor exposed." *International Transactions in Operational Research* 22.1 (2015): 3-18.